FACULTY WORKING PAPER 93-0137

Forecasting Exchange Rates Using Feedforward and Recurrent Neural Networks - Revised

Chung-Ming Kuan
Department of Economics
University of Illinois

Tung Liu
Department of Economics
Ball State University

Bureau of Economic and Business Research
College of Commerce and Business Administration
University of Illinois at Urbana-Champaign

BEBR

FACULTY WORKING PAPER NO. 93-0137

College of Commerce and Business Administration

University of Illinois at Urbana-Champaign

May 1993

Forecasting Exchange Rates Using Feedforward and Recurrent Neural Networks - Revised

Chung-Ming Kuan
Tung Liu

FORECASTING EXCHANGE RATES USING FEEDFORWARD AND RECURRENT NEURAL NETWORKS

Chung-Ming Kuan

Department of Economics
University of Illinois at Urbana-Champaign

and

Tung Liu

Department of Economics
Ball State University

First Draft: May 15, 1992
Revised: April 20, 1993

† We would like to thank Roger Koenker, Bill Maloney, Paul Newbold, and four anonymous referees for their invaluable suggestions and comments. We are most grateful to Richard Baillie for providing us the data set and to Bruce Mizrach and Hal White for permitting us to access their programs. C.-M. Kuan also thanks the Research Board of the University of Illinois for research support. All remaining errors are ours. An early version of this paper (based on a different data set) was presented at the 1992 North American Winter Meeting of the Econometric Society in New Orleans, Louisiana.

Summary

In this paper we investigate the forecasting ability of feedforward and recurrent neural networks based on empirical foreign exchange rate data. A two-step procedure is proposed to construct suitable networks, in which networks are selected based on the predictive stochastic complexity (PSC) criterion, and the selected networks are estimated using both recursive Newton algorithms and the method of nonlinear least squares. We find that PSC is a sensible criterion for selecting networks and that the out-of-sample performance of neural networks is reasonably good. In particular, the networks selected based on PSC have rather satisfactory out-of-sample sign prediction results, in contrast with some commonly used ARMA models.

1 Introduction

Neural networks provide a general class of nonlinear models which has been successfully applied in many different fields. Numerous empirical and computational applications can be found in the Proceedings of the International Joint Conference on Neural Networks and Conference of Neural Information Processing Systems. In spite of its success in various fields, there are only a few applications of neural networks in economics. Neural networks are novel in econometric applications in the following two respects. First, the class of multilayer neural networks can well approximate a large class of functions (Hornik, Stinchcombe, and White (1989) and Cybenko (1989)), whereas most of commonly used nonlinear time-series models do not have this property. Second, as shown in Barron (1991), neural networks are more parsimonious models than linear subspace methods such as polynomial, spline, and trigonometric series expansions in approximating unknown functions. Thus, if the behavior of economic variables exhibits nonlinearity, a suitably constructed neural network can serve as a useful tool to capture such regularity.

In this paper we investigate possible nonlinear patterns in foreign exchange data using *feedforward* and *recurrent* networks. It has been widely accepted that foreign exchange rates are I(1) (integrated of order one) processes and that changes of exchange rates are uncorrelated over time. Hence, changes in exchange rates are not linearly predictable in general. For a comprehensive review of these issues, see Baillie and McMahon (1989). Since the empirical studies supporting these conclusions rely mainly on linear time series techniques, it is not unreasonable to conjecture that the linear unpredictability of exchange rates may be due to limitations of linear models. Hsieh (1989) finds that changes of exchange rates may be nonlinearly dependent, even though they are linearly uncorrelated. Some researchers also provide evidence in favor of nonlinear forecasts, e.g., Taylor (1980,1982), Engel and Hamilton (1990), Engel (1991), and Chinn (1991). On the other hand, Diebold and Nason (1990) find that nonlinearities of exchange rates, if any, cannot be exploited to improve forecasting. Therefore, we treat neural networks as alternative nonlinear models and focus on whether neural networks can provide superior out-of-sample forecasts.

This paper has two objectives. First, we introduce different neural network modeling techniques and propose a two-step procedure to construct suitable neural networks. Second, we evaluate the performance of networks obtained from the proposed procedure in terms of out-of-sample MSE (mean squared errors) and sign predictions (i.e., forecasts of the direction of future changes). In the first step of the proposed procedure, we apply recursive Newton algorithms to estimate networks and compute the so-called "predictive stochastic complexity" (Rissanen (1987)), from which we can easily select suitable networks. In the second step, statistically more efficient estimates are obtained by the method of nonlinear least squares using recursive estimates from the first step as initial values. Our procedure differs from previous applications of feedforward networks in economics, e.g., White (1988) and Kuan and White (1990), in that networks are selected objectively. Also, the application of recurrent networks is new in applied econometrics; hence its performance should also be of interest to researchers. Our results show that predictive stochastic complexity is a sensible criterion for selecting networks and that the resulting networks perform reasonably well in different out-of-sample periods. In particular, the selected networks yield quite satisfactory out-of-sample sign predictions which are significantly better than predictions based on tossing a coin. This result is in contrast with that of some commonly used ARMA models.

This paper proceeds as follows. We review various network architectures and estimation methods in section 2. The network construction procedures are described in section 3. Empirical results are analyzed in section 4. Section 5 concludes the paper.

2 Feedforward and Recurrent Networks

In this section we briefly describe feedforward and recurrent networks and associated estimation methods. For more details see Kuan and White (1993a).

2.1 Network Functional Forms

A neural network may be interpreted as a nonlinear regression function characterizing the relationship between the dependent variable (target) y and an n-vector of explanatory

variables (inputs) x. Instead of postulating a specific nonlinear function, a neural network model is constructed by combining many "basic" nonlinear functions via a multi-layer structure. In a *feedforward* network, the explanatory variables first simultaneously activate q hidden units in an intermediate layer through some function Ψ, and the resulting hidden-unit activations h_i, $i = 1, \cdots, q$, then activate output units through some function Φ to produce the network output o (see Figure 1). Symbolically, we have

$$\begin{aligned} h_{i,t} &= \Psi(\gamma_{i0} + \sum_{j=1}^{n} \gamma_{ij} x_{j,t}), \qquad i = 1, \cdots, q, \\ o_t &= \Phi(\beta_0 + \sum_{i=1}^{q} \beta_i h_{i,t}), \end{aligned} \tag{1}$$

or more compactly,

$$\begin{aligned} o_t &= \Phi\Big(\beta_0 + \sum_{i=1}^{q} \beta_i \Psi(\gamma_{i0} + \sum_{j=1}^{n} \gamma_{ij} x_{j,t})\Big) \\ &=: f(x_t, \theta), \end{aligned} \tag{2}$$

where θ is the vector of parameters containing all β's and γ's. This is a flexible nonlinear functional form in that the activation functions Ψ and Φ can be chosen quite arbitrarily, except that Ψ is generally required to be a bounded function. Hornik, Stinchcombe, and White (1989) and Cybenko (1989) show that the function f constructed in (2) can approximate a large class of functions arbitrarily well (in a suitable metric), provided that the number of hidden units, q, is sufficiently large. This property is very similar to that of nonparametric methods. Barron (1991) also shows that a feedforward network can achieve an approximation rate $O(1/q)$ by using a number of parameters $O(qn)$ that grows linearly in q, whereas traditional polynomial, spline, and trigonometric expansions require exponentially $O(q^n)$ terms to achieve the same approximation rate. Thus, neural networks are relatively more parsimonious than these series expansion in approximating unknown functions. These two properties make feedforward networks an attractive econometric tool in (nonparametric) applications.

[Figure 1 About Here]

In a dynamic context, it is natural to include lagged dependent variables as explanatory variables in a feedforward network to capture dynamics. This approach suffers the

drawback that the correct number of lags needed is typically unknown (this is analogous to the problem of determining the order of an autoregression). Hence, the lagged dependent variables in a network may not be enough to characterize the behavior of y in some applications. To overcome this deficiency, various *recurrent* networks, i.e., networks with feedbacks, have been proposed. A recurrent network has a richer dynamic structure and is similar to a linear time-series model with moving average terms. In particular, we consider the following network due to Elman (1990) (see Figure 2):

$$\begin{aligned} h_{i,t} &= \Psi(\gamma_{i0} + \sum_{j=1}^{n} \gamma_{ij} x_{j,t} + \sum_{\ell=1}^{q} \delta_{i\ell} h_{\ell,t-1}) \\ &=: \psi_i(x_t, h_{t-1}, \theta), \qquad i = 1, \cdots, q, \\ o_t &= \Phi(\beta_0 + \sum_{i=1}^{q} \beta_i \psi_i(x_t, h_{t-1}, \theta)) \\ &=: \phi(x_t, h_{t-1}, \theta), \end{aligned} \tag{3}$$

where θ denotes the vector of parameters containing all β's, γ's, and δ's. Here, the hidden-unit activations h_i feed back to the input layer with delay and serve to "memorize" the past information, cf. (1). From (3) we can write, by recursive substitution,

$$h_{i,t} = \psi_i(x_t, \psi_i(x_{t-1}, h_{t-2}, \theta), \theta) = \cdots =: r_i(x^t, \theta), \qquad i = 1, \cdots, q, \tag{4}$$

where $x^t = (x_t, x_{t-1}, \cdots, x_1)$. Hence, $h_{i,t}$ depends on x_t and its entire history. It follows that

$$o_t = \phi(x_t, h_{t-1}, \theta) =: g(x^t, \theta) \tag{5}$$

is also a function of x_t and its entire history, cf. (2). In view of (5), we expect that a recurrent network may capture more dynamic characteristics of y_t than does a feedforward network.

[Figure 2 About Here]

2.2 Estimation Methods

Given a dependent variable y and a feedforward network (2) with explanatory variables x, we want to find suitable parameters θ^* minimizing

$$E\,|y - f(x,\theta)|^2 = E\,|y - E(y|x)|^2 + E\,|E(y|x) - f(x,\theta)|^2. \tag{6}$$

This is equivalent to minimizing $E\,|E(y|x) - f(x,\theta)|^2$. That is, we want to use the feedforward network to approximate the unknown conditional mean function and minimize the resulting squared approximation errors. Since $E(y|x)$ is the best L_2-predictor of y given x, the network output $o_t = f(x_t, \theta^*)$ should match y_t fairly closely, at least in the L_2 sense. In view of (6), the unknown parameters can be estimated using the method of Nonlinear Least Squares (NLS). Alternatively, recursive estimation methods may be used. Although recursive estimation is important for adaptive learning and on-line signal processing, it is well known that recursive algorithms do not utilize the data efficiently in finite samples. However, recursive estimation can provide useful starting values for the NLS estimator and facilitate network selection (see discussions in Section 3). Specifically, we consider the following *stochastic Newton* algorithm:

$$\begin{aligned}\hat{\theta}_{t+1} &= \hat{\theta}_t + \eta_t \hat{G}_t^{-1} \nabla f(x_t, \hat{\theta}_t)[y_t - f(x_t, \hat{\theta}_t)], \\ \hat{G}_{t+1} &= \hat{G}_t + \eta_t [\nabla f(x_t, \hat{\theta}_t) \nabla f(x_t, \hat{\theta}_t)' - \hat{G}_t],\end{aligned} \tag{7}$$

where $\nabla f(x,\theta)$ is the (column) gradient vector of f with respect to θ and $\{\eta_t\}$ is a sequence of learning rates of order $1/t$. Note that $\nabla f(x,\theta)[y - f(x,\theta)]$ is the vector of the first-order derivatives of the squared-error loss: $[y - f(x,\theta)]^2$ and that the second updating equation recursively estimates an approximate Newton direction. Thus, the algorithm (7) perform a recursive Newton search in the parameter space. Kuan and White (1993a) show that the estimates of (7) are root-T consistent and asymptotically equivalent to the NLS estimator under very general conditions. In practice, an algebraically equivalent form of (7) which does not involve matrix inversion can be used to simplify computation, see Kuan and White (1993a). We also note that if f is a linear function, the algorithm (7) reduces to the well-known recursive least square algorithm, see e.g., Ljung and Söderström (1983).

Similarly, the parameters of interest of a recurrent network are θ^* that minimize

$$E\,|y_t - g(x^t, \theta)|^2.$$

Here, $g(x^t, \theta^*)$ can be viewed as an approximation of $E(y_t|x^t)$. In view of (4) and (5), h_t and o_t depend on θ directly and indirectly through the presence of lagged hidden-unit activations h_{t-1}; hence both r and g are complex functions of θ. In particular, in

calculating the derivatives of g with respect to θ, parameter dependence of h_{t-1} must be taken into account. Owing to this "state dependent" structure, it is difficult to implement the method of NLS, and the algorithm (7) is invalid.

A recurrent Newton algorithm analogous to (7) is

$$\begin{aligned}
\hat{e}_t &= y_t - \phi(x_t, \hat{h}_{t-1}, \hat{\theta}_t), \\
\nabla \hat{e}_t &= -\phi_\theta(x_t, \hat{h}_{t-1}, \hat{\theta}_t) - \hat{\Delta}_t\, \phi_h(x_t, \hat{h}_{t-1}, \hat{\theta}_t), \\
\hat{\theta}_{t+1} &= \hat{\theta}_t - \eta_t\, \hat{G}_t^{-1}\, \nabla \hat{e}_t\, \hat{e}_t, \\
\hat{G}_{t+1} &= \hat{G}_t + \eta_t\, (\nabla \hat{e}_t \nabla \hat{e}_t' - \hat{G}_t),
\end{aligned} \tag{8}$$

where the i-th ($i = 1, \ldots, q$) hidden-unit activation is updated according to

$$\hat{h}_{i,t} = \Psi\Big(\hat{\gamma}_{i0,t} + \sum_{j=1}^{n} \hat{\gamma}_{ij,t} x_{j,t} + \sum_{\ell=1}^{q} \hat{\delta}_{i\ell,t} \hat{h}_{\ell,t-1}\Big) = \psi_i(x_t, \hat{h}_{t-1}, \hat{\theta}_t), \tag{9}$$

the j-th ($j = 1, \ldots, q$) column of $\hat{\Delta}_{t+1}$ is updated according to

$$\hat{\Delta}_{j,t+1} = \psi_{j,\theta}(x_t, \hat{h}_{t-1}, \hat{\theta}_t) + \hat{\Delta}_t\, \psi_{j,h}(x_t, \hat{h}_{t-1}, \hat{\theta}_t), \tag{10}$$

and the initial values $\hat{\theta}_0$, $\hat{h}_0$, and $\hat{\Delta}_0$ are chosen arbitrarily. Here, ϕ_θ and ϕ_h are column vectors of the first order derivatives of ϕ with respect to θ and h, respectively, and $\psi_{j,\theta}$ and $\psi_{j,h}$ are column vectors of the first order derivatives of the j-th hidden unit ψ_j with respect to θ and h, respectively. The recurrent Newton algorithms differs from (7) in that updating equations (9) and (10) allow us to update the $dh_{t-1}/d\theta$ term recursively. Clearly, a recurrent network not depending on h_{t-1} is a feedforward network. In this case, the ϕ_h term is zero so that the updating equations of $\hat{\Delta}_t$ are not needed, and (8) simply reduces to the standard Newton algorithm (7). In view of the first equation in (3), we can see that certain constraints must be imposed to prevent h from being "explosive". Kuan (1993) shows that the recurrent Newton algorithm is strongly consistent, provided that $|\delta_{i\ell}| < 4/q$ for all i and ℓ, where q is the number of hidden units, and is computationally more efficient than the "recurrent back-propagation" algorithm of Kuan, Hornik, and White (1993); see also Kuan and White (1993b).

3 Network Construction

In this paper, we choose the activation functions Ψ as the logistic function and Φ as the identity function in the networks (1) and (3). These choices are quite standard in the neural network literature. Our dependent variables are changes of log exchange rates, and for each exchange rate, networks are constructed using lagged dependent variables as explanatory variables. The resulting networks are therefore nonlinear AR models.

A difficult problem in network construction is to determine network complexity. This involves the determination of the number of lagged dependent variables and the number of hidden units. A very simple network may not be able to approximate the unknown conditional mean function well; an excessively complex network may over fit the data. There is, however, no definite conclusion regarding the determination of network complexity. As neural network models are, by construction, some approximating functions, it is our opinion that the determination of network complexity is a model selection problem. One possible criterion is the Schwarz (1978) Information Criterion (SIC). Rissanen (1983,1984) show that this criterion can be applied to a more general setting than linear models; in particular, the SIC is asymptotically equivalent to *stochastic complexity* of a model (Rissanen (1987)). Note, however, that selecting networks based on SIC is computationally demanding because NLS is required for estimating *every* possible network.

An alternative criterion to regularize network complexity is the "Predictive Stochastic Complexity" (PSC) criterion due to Rissanen (1986a,b); see also Rissanen (1987). Given a function $m(x, \theta)$, where θ is a k-dimensional parameter vector, and a sample of T observations, PSC is computed as the average of squared, "honest", prediction errors:

$$\frac{1}{T-k} \sum_{t=k+1}^{T} (y_t - m(x_t, \hat{\theta}_t))^2, \tag{11}$$

where $\hat{\theta}_t$ is the predicted parameter obtained from the data up to time $t-1$. The prediction error $y_t - m(x_t, \hat{\theta}_t)$ is "honest" in the sense that no information at time t or beyond is used to calculate $\hat{\theta}_t$. A particular model is selected if it has the smallest PSC within a class of models. If two models have the same PSC, the simpler one is selected. Clearly, the PSC criterion is based on forward validation, which is particularly important in forecasting.

Rissanen also shows that for encoding a sequence of numbers, the PSC criterion can determine the code with the shortest code length asymptotically. For a thorough discussion of the notion of stochastic complexity we refer to Rissanen (1989). Obviously, calculation of PSC is also computationally demanding if NLS is required to estimate $\hat{\theta}_t$ at each t. Following the idea of Gerencsér and Rissanen (1992), we can compute $\hat{\theta}_t$ using recursive estimation methods, which are more tractable computationally. Clearly, both (7) and (8) can be easily applied to compute PSC.

We therefore adopt the following two-step procedure to construct suitable networks.

1. Recursive estimation. A family of networks with different numbers of explanatory variables and hidden units is estimated using the stochastic Newton algorithm (7) or the recurrent Newton algorithm (8).

   (a) Ten sets of initial parameters are generated randomly from $N(0, 1)$, and the one that results in the lowest MSE is used as the initial values for recursive algorithms.

   (b) We let the algorithm run through the data set once and compute the PSC values. The three best networks according to the PSC values are selected.

2. NLS estimation. The FORTRAN subroutine LMDER in MINPACK[1] is used.

   (a) For selected feedforward networks, the final recursive estimates are used as initial values of the NLS estimator for θ.

   (b) For selected recurrent networks, we fix the recurrent parameters, δ's, at the final recursive estimates and use the rest of the recursive estimates as initial values of the NLS estimator for the parameters β's and γ's.

In the proposed procedure, both recursive and NLS estimations are used. Recursive estimation facilitates network selection because PSC can be easily computed using the Newton

[1] MINPACK is a collection of FORTRAN subroutines from Argonne National Laboratory, and LMDER is one of its NLS subroutines. LMDER is based on a modification of the Levenberg-Marquardt algorithm; details of this algorithm can be found in More (1977).

algorithms and is particularly important for recurrent networks. Moreover, the recursive estimates may provide useful starting values of the NLS estimator in the next step. NLS estimation in the second step is used to improve efficiency of parameter estimates. This two-step estimation is analogous to that of White (1989). Note that the parameters δ's in recurrent networks are fixed in the second step to avoid constraint minimization. (Recall that δ's must be constrained suitably to ensure proper convergence behavior.) Hence, the second step for recurrent network construction is analogous to building a partially hard-wired recurrent network (Kuan and Hornik (1991)).

4 Empirical Results

In this paper five exchange rates against the U.S. dollar, including British Pound (BP), Canadian Dollar (CD), Deutsche Mark (DM), Japanese Yen (JY), and Swiss Franc (SF), are investigated. The data are daily opening bid prices of the NY Foreign Exchange Market from March 1, 1980 to January 28, 1985, consisting of 1245 observations. All series except BP are US dollars per unit of foreign currency. This data set has also been used in Baillie and Bollerslev (1989). Let $S_{i,t}$ denote the i-th exchange rate at time t, and $y_{i,t} = \log S_{i,t} - \log S_{i,t-1}$, i = BP, CD, DM, JY, SF. By applying various unit-root tests, Baillie and Bollerslev (1989) find that $\log S_{i,t}$ are unit root processes without drift and that $y_{i,t}$ behave like a martingale difference sequence. We also estimated thirty six ARMA models for $y_{i,t}$ from ARMA(0,0) to ARMA(5,5) and found that ARMA(0,0) is the best model for all five series in terms of the SIC values. This is consistent with the results of Baillie and Bollerslev (1989). In what follows, we will abuse terminology and refer to ARMA(0,0) as the random walk model.

To evaluate the forecasting performance of different models of $y_{i,t}$, we reserve the last 50, 100, and 150 observations as out-of-sample periods and estimate models using 1194, 1144, and 1094 observations, respectively. These choices are arbitrary. Of particular interest to us is whether a model can outperform the random walk model in terms of out-of-sample MSE. We apply the Mizrach (1992) test[2] to check whether the out-of-sample

[2]These tests are computed based on the program provided by Prof. Mizrach. In our computation, models

MSE of network models are significantly different from those of the random walk model. In addition, we evaluate out-of-sample sign predictions of $y_{i,t}$. Sign prediction gives forecasts of the direction of future changes and yields important information in financial forecasting. In an extreme case, a model could have small out-of-sample MSE but predict all the signs incorrectly, and hence be virtually useless. We expect a good model to have the proportion of correct sign predictions (in out-of-sample period) significantly better than 0.5, i.e., better than predictions based on tossing a coin. Thus, taking the null hypothesis as $p = 0.5$, the following test statistic is used:

$$\sqrt{n}(\bar{z} - p)/\sqrt{p(1-p)} \;=\; \sqrt{n}(\bar{z} - 0.5)/0.5 \;\longrightarrow^{d}\; N(0,1),$$

where $\bar{z}$ is the proportion of correct sign predictions and n is the number of observations in an out-of-sample period. (We thank a referee for this suggestion.) For a one-sided test and $n = 50$, 100, and 150, it is easily verified that at the 5% level, $\bar{z} > .616$, .583, and .567 is significant, and that at the 10% level, $\bar{z} > .591$, .564, and .552 is significant.

Neural network models are constructed according to the two-step procedure described in Section 3. Note that for each series, the network explanatory variables are lagged dependent variables[3]. In the first step, thirty feedforward and recurrent networks (with 1–6 lagged $y_{i,t}$ and 2–6 hidden units) are estimated using the recursive Newton algorithms, and the three networks with best PSC values are selected. In the second step, the selected networks are further "smoothed" using the method of NLS. (We omit networks with one hidden unit because they are not practically interesting.) Out-of-sample forecasting results from recursive and NLS estimation are summarized in Tables 1–5, where we write the network with L lags and H hidden units as the network (L, H). Ideally, we can construct a multiple -output network for all 5 series, analogous to a multivariate nonlinear regression model. A program implementing multiple-output networks is currently under

with MSE smaller than the random walk model have positive statistics. As the limiting distribution of this statistic is $N(0,1)$, the critical values of an one-sided test at 5% and 10% levels are 1.645 and 1.282, respectively.

[3]We have also constructed networks for each $y_{i,t}$ using lagged $y_{j,t}$, $j \neq i$, as additional explanatory variables. The results are not particularly exciting. We therefore confine ourselves to networks of the present form which, as we have mentioned, are simply nonlinear AR models.

development.

[Tables 1–5 About Here]

We first observe that a wide variety of networks have been selected and that for each series, selected networks have quite similar structures. In particular, there is at least one common (feedforward or recurrent) network selected in three (or two) in-sample periods. These common networks are[4]:

1. BP: feedforward (5,3); recurrent (1,2) and (6,2).
2. CD: feedforward (1,4); recurrent (1,2) and (2,2).
3. DM: feedforward (2,2); recurrent (1,2) and (4,2).
4. JY: feedforward (1,3); recurrent (1,3).
5. SF: feedforward (2,2); recurrent (1,2) and (1,4).

Note that the structures of these common networks are not very complex. These results seem to suggest that there exists only mild nonlinearity in these series.

We also observe the following.

1. In terms of out-of-sample MSE:
   (a) All selected networks are *not* significantly better (or worse) than the random walk model.
   (b) Selected feedforward and recurrent networks do not dominate each other; NLS forecasting results need not be better than corresponding recursive results.
   (c) Except for the CD, all the common networks listed above have out-of-sample MSE (from recursive results) smaller than those of the random walk model.
2. In terms of out-of-sample sign predictions:

[4] For the JY and feedforward networks of the BP, the common networks listed are taken from the periods with 100 and 150 test observations.

(a) Except for the CD, many selected networks have correct sign predictions about 60% and are significantly better than tossing a coin.

(b) Selected feedforward and recurrent networks do not dominate each other; NLS sign predictions need not be better than corresponding recursive results.

(c) Except for the CD, all the common networks have out-of-sample sign predictions (from recursive results) better than tossing a coin. Most of these recursive prediction results are also better than the corresponding NLS results.

These results show that the PSC criterion is a quite sensible criterion to determine network structure. The results for out-of-sample MSE suggest that the "captured" nonlinearity cannot be exploited to improve forecasting MSE. The results for sign predictions seem to indicate that there may be some hope of predicting the directions of future changes based on recursive estimation results. In fact, obtaining correct sign predictions consistently about 60% of the time (or more) in four out of five series is quite encouraging. We note that our estimation methods are based on MSE minimization, which is not a loss function specific for sign predictions. It would be interesting to construct estimation methods based on a suitable loss function; this is beyond the scope of this paper, however. Although the performance of recurrent networks is different from that of feedforward networks, it is somewhat surprising to us that recurrent networks do not outperform feedforward networks. One possible interpretation is that the feedback structure in recurrent networks cannot be very effective when there is very little correlation across the dependent variables.

For the sake of comparison, we also evaluate the out-of-sample performance of four commonly used ARMA models[5], including ARMA(1,0), (0,1), (1,1), and (2,2). The results are summarized in Table 6. For the JY, ARMA models have out-of-sample MSE significantly better than those of the random walk model in two forecasting periods with

[5] A referee points out that a more comparable way is to select and estimate ARMA models also based on the proposed two-step procedure. Selecting ARMA models based on PSC has been discussed by, e.g., Gerencsér (1990). However, implementing the two-step procedure is more involved, see e.g., Ljung and Söderström (1983). As our emphasis is on neural network models, we do not pursue this possibility.

100 and 150 test observations, but this dominance disappears in the period with 50 observations. For the CD, ARMA models are significantly worse than the random walk model in the forecasting period with 150 test observations but become significantly better in the forecasting period with 50 observations. For other series, ARMA models forecasts are not significantly different from those of the random walk model. We also observe that most of the correct sign predictions of ARMA models fluctuate around 50%, except those of ARMA(1,0) and (0,1) for the BP.

[Table 6 About Here]

5 Conclusions

In this paper we propose a two-step procedure to estimate and select feedforward and recurrent networks and carefully evaluate the forecasting performance of selected networks in different out-of-sample periods. We find that PSC is a sensible criterion in selecting networks. Based on this criterion, it is possible to find a network with better out-of-sample MSE and/or sign predictions, compared with the random walk model. Hence, the proposed two-step procedure may be used as a standard network construction procedure in other applications. Our results show that these networks are not significantly better than the random walk model in terms of out-of-sample MSE, however. Therefore, we confirm the conclusion of Diebold and Nason (1990) that nonlinearity of exchange rates may not be exploited to improve point forecasts. If we are not so ambitious about point forecasts and confine ourselves to sign predictions, our results also suggest that network models perform quite well for this purpose in four out of five series we investigated. In particular, selected networks have sign predictions systematically better than predictions based on coin tossing. This is an interesting direction for further research. On the other hand, ARMA models may have out-of-sample MSE significantly better than the random walk model, but their correct sign predictions are typically fluctuating around 50%. Finally, our results show that there is no significant difference between feedforward and recurrent networks in this application.

References

Baillie, R. T. and T. Bollerslev (1989), 'Common Stochastic Trends in a System of Exchange Rates', *Journal of Finance*, **44**, 167–181.

Baillie, R. T. and P. C. McMahon (1989), *The Foreign Exchange Market: Theory and Econometric Evidence*, Cambridge University Press, New York.

Barron, A. R. (1991), 'Universal approximation bounds for superpositions of a sigmoidal function', Technical Report No. 58, Department of Statistics, University of Illinois, Urbana-Champaign.

Chinn, M. D. (1991), 'Some linear and nonlinear thoughts on exchange rates', *Journal of International Money and Finance*, **10**, 214–230.

Cybenko, G. (1989), 'Approximations by superpositions of a sigmoidal function', *Mathematics of Control, Signals and Systems*, **2**, 303–314.

Diebold, F. X. and J. A. Nason (1990), 'Nonparametric exchange rate prediction?', *Journal of International Economics*, **28**, 315–332.

Elman, J. L. (1990), 'Finding structure in time', *Cognitive Science*, **14**, 179–211.

Engel, C. (1991), 'Can the Markov switching model forecast exchange rates?', Working Paper, University of Washington.

Engel, C. and J. Hamilton (1990), 'Long Swings in the exchange rates: Are they in the data and do markets know it?', *American Economic Review*, **80**, 689–713.

Gerencsér, L. (1990), 'On Rissanen's predictive stochastic complexity for stationary ARMA processes', Technical Report, Department of Electrical Engineering, McGill University.

Gerencsér, L. and J. Rissanen (1992), 'Asymptotics of predictive stochastic complexity', in D. Brillinger, P. Caines, J. Geweke, E. Parzen, M. Rosenblatt, and M. Taqqu

(ed.), *New Directions in Time Series Analysis*, Part 2, Springer-Verlag, New York.

Hornik, K., Stinchcombe, M., and H. White (1989), 'Multi-layer feedforward networks are universal approximators', *Neural Networks*, **2**, 359–366.

Kuan, C.-M. (1993), 'A Recurrent Newton Algorithm and Its Convergence Property', in preparation.

Kuan, C.-M. and K. Hornik (1991), 'Learning in a partially hard-wired recurrent network', *Neural Network World*, **1**, 39–45.

Kuan, C.-M., Hornik, K. and H. White (1993), 'A convergence result for learning in recurrent neural networks', Discussion Paper, Department of Economics, University of California, San Diego.

Kuan, C.-M. and H. White (1990), 'Predicting appliance ownership using logit, neural network, and regression tree models', BEBR Working Paper 90-1647, College of Commerce, University of Illinois, Urbana-Champaign.

Kuan, C.-M. and H. White (1993a), 'Artificial neural networks: An econometric perspective', *Econometric Reviews*, forthcoming.

Kuan, C.-M. and H. White (1993b), 'Adaptive learning with nonlinear dynamics driven by dependent processes', Discussion Paper, Department of Economics, University of California, San Diego.

Ljung, L. and T. Söderström (1983), *Theory and Practice of Recursive Identification*, MIT Press, Cambridge.

Mizrach, B. (1992), 'Forecast comparison in L_2', Working paper, University of Pennsylvania.

More, J. (1977), 'The Levenberg-Marquardt algorithm, implementation and theory', in G. A. Watson (ed.), *Numerical Analysis*, Springer-Verlag, New York.

Rissanen, J. (1983), 'A universal prior for integers and estimation by minimum description

length', *Annals of Statistics*, **11**, 416–431.

Rissanen, J. (1984), 'Universal coding, information, prediction, and estimation', *IEEE Transactions on Information Theory*, **IT-30**, 629–636.

Rissanen, J. (1986a), 'A predictive least-squares principle', *IMA Journal of Mathematical Control & Information*, **3**, 211–222.

Rissanen, J. (1986b), 'Stochastic complexity and modeling', *Annals of Statistics*, **14**, 1080–1100.

Rissanen, J. (1987), 'Stochastic complexity (with discussions)', *Journal of the Royal Statistical Society*, B, **49**, 223-239 and 252–265.

Rissanen, J. (1989), *Stochastic Complexity in Statistical Inquiry*, World Science Publishing Co, Singapore.

Schwarz, G. (1978), 'Estimating the dimension of a model', *Annals of Statistics*, **6**, 461–464.

Taylor, S. J. (1980), 'Conjectured models for trends in financial prices, tests and forecasts', *Journal of the Royal Statistical Society*, A, **143**, 338–362.

Taylor, S. J. (1982), 'Tests of the random walk hypothesis against a price trend hypothesis', *Journal of Financial and Quantitative Analysis*, **17**, 37–61.

White, H. (1988), 'Economic prediction using neural networks: The case of IBM Stock Prices', in *Proceedings of the IEEE Second International Conference on Neural Networks*, II, pp. 451–458, SOS Printing, San Diego.

White, H. (1989), 'Some asymptotic results for learning in single hidden-layer feedforward network models', *Journal of the American Statistical Association*, **84**, 1003–1013.

Table 1. Out-of-Sample MSE and Sign Predictions from Selected Networks: British Pound.

Network Type	Test Obs.	Selected Network	Recursive Result			NLS Result	
			PSC	MSE	Sign	MSE	Sign
Feed-forward	50	(1,2)	.4359	.3566 (.6101)	64.0**	.3657 (.6006)	62.0**
		(1,4)	.4363	.3652 (.6180)	72.0**	.3627 (.5957)	58.0
		(2,2)	.4365	.3656 (.6324)	72.0**	.3822 (.5506)	72.0**
	100	(5,3)	.4210	.5575 (.6114)	62.0**	.5915 (-.5283)	54.0
		(4,3)	.4211	.5956 (-.5645)	59.0**	.6130 (-.6062)	40.0
		(6,2)	.4211	.5637 (.5654)	62.0**	.5588 (.5392)	61.0**
	150	(5,3)	.4242	.4930 (.5701)	62.0**	.5028 (-.4124)	54.7
		(4,3)	.4244	.5146 (-.5506)	56.7**	.5354 (-.6166)	40.7
		(1,2)	.4246	.4859 (.6279)	59.3**	.4820 (.6418)	59.3**
Recurrent	50	(6,2)	.4352	.3672 (.6069)	66.0**	.3714 (.5666)	58.0
		(1,2)	.4360	.3661 (.6325)	72.0**	.3708 (.6153)	72.0**
		(2,4)	.4365	.3597 (.6290)	72.0**	.3697 (.6083)	68.0**
	100	(1,2)	.4201	.5631 (.5699)	60.0**	.5568 (.6146)	61.0**
		(6,2)	.4209	.5632 (.5656)	61.0**	.5712 (.4300)	59.0**
		(4,4)	.4210	.5839 (-.5410)	55.0	.6007 (-.5609)	54.0
	150	(1,2)	.4231	.4930 (.5868)	58.7**	.4872 (.6268)	59.3**
		(6,2)	.4236	.4997 (.4690)	59.3**	.5149 (-.5731)	46.7
		(6,3)	.4236	.5041 (-.4623)	58.7**	.5220 (-.5688)	58.7**

Notes: The selected networks are ordered from the best to the 3rd best, according to the PSC values. "MSE" stands for out-of-sample MSE; "Sign" stands for the proportions of correct sign predictions in out-of-sample periods. The numbers in the parentheses in MSE columns are Mizrach's MSE-comparison statistics. For sign prediction results, * and ** stand for significance at 10% and 5% level, respectively. The other tables follow the same convention.

Table 2. Out-of-Sample MSE and Sign Predictions from Selected Networks: Canadian Dollars.

Network Type	Test Obs.	Selected Network	Recursive Result			NLS Result	
			PSC	MSE	Sign	MSE	Sign
Feed-forward	50	(1,4)	.6134	.1882 (.4814)	54.0	.1887 (.4826)	56.0
		(1,5)	.6152	.1888 (.4674)	54.0	.1937 (-.5071)	56.0
		(1,3)	.6173	.1885 (.5030)	56.0	.1885 (.4749)	54.0
	100	(1,4)	.6218	.3162 (-.5203)	49.0	.3134 (-.4740)	52.0
		(5,2)	.6247	.3301 (-.6273)	44.0	.3514 (-.6163)	52.0
		(2,2)	.6253	.3137 (-.4619)	49.0	.3085 (.4830)	53.0
	150	(1,4)	.6221	.4167 (-.4417)	49.3	.4155 (.4039)	52.0
		(2,2)	.6251	.4190 (-.4917)	48.0	.4145 (.4525)	50.0
		(1,2)	.6254	.4199 (-.5248)	47.3	.4162 (-.3945)	51.3
Recurrent	50	(1,3)	.6128	.1880 (.5009)	56.0	.1914 (-.4193)	52.0
		(2,2)	.6148	.1883 (.5422)	56.0	.1879 (.5061)	56.0
		(1,2)	.6161	.1860 (.5546)	56.0	.1876 (.5206)	56.0
	100	(2,2)	.6243	.3121 (-.4156)	51.0	.3114 (.4339)	51.0
		(5,2)	.6276	.3103 (.4980)	52.0	.3083 (.5606)	51.0
		(1,2)	.6277	.3117 (.3881)	51.0	.3111 (.4568)	52.0
	150	(2,2)	.6242	.4175 (-.4941)	49.3	.4162 (-.4020)	49.3
		(1,2)	.6276	.4198 (-.5073)	48.7	.4158 (.3275)	50.7
		(5,2)	.6276	.4158 (.3523)	50.0	.4141 (.4912)	49.3

Note: PSC and MSE are the numbers in the table $\times 10^{-1}$.

Table 3. Out-of-Sample MSE and Sign Predictions from Selected Networks: Deutsche Mark.

Network	Test	Selected	Recursive Result			NLS Result	
Type	Obs.	Network	PSC	MSE	Sign	MSE	Sign
Feed-forward	50	(2,2)	.4983	.1989 (.5805)	62.0^{**}	.1895 (.5592)	52.0
		(5,2)	.4988	.1994 (.6063)	60.0^{*}	.1999 (.5554)	64.0^{**}
		(2,5)	.4995	.1963 (.6172)	64.0^{**}	.1942 (.5769)	64.0^{**}
	100	(2,5)	.4741	.5969 (.5603)	61.0^{**}	.7917 (-.5657)	58.0^{*}
		(2,2)	.4758	.5976 (.5606)	60.0^{**}	.6060 (-.3605)	52.0
		(2,4)	.4760	.5962 (.5705)	60.0^{**}	.6008 (.4831)	57.0^{*}
	150	(5,2)	.4809	.5236 (.5064)	58.0^{**}	.5333 (-.4856)	50.7
		(2,2)	.4810	.5202 (.5728)	58.0^{**}	.5339 (-.4954)	53.3
		(4,2)	.4815	.5233 (.5475)	61.3^{**}	.5419 (-.5922)	58.0^{**}
Recurrent	50	(1,2)	.4976	.2006 (.6028)	62.0^{**}	.2014 (.5989)	62.0^{**}
		(4,2)	.4998	.1989 (.5827)	62.0^{**}	.2030 (.5024)	60.0^{*}
		(5,2)	.4999	.1993 (.5895)	60.0^{*}	.1943 (.5680)	62.0^{**}
	100	(1,2)	.4734	.6014 (.5115)	61.0^{**}	.5915 (.5629)	55.0
		(4,2)	.4760	.6043 (.4391)	60.0^{**}	.6120 (-.4826)	56.0
		(1,4)	.4767	.6066 (-.4335)	59.0^{**}	2.088 (-.5628)	50.0
	150	(5,2)	.4791	.5297 (-.4756)	58.7^{**}	.5441 (-.5712)	57.3^{**}
		(1,2)	.4793	.5210 (.5869)	60.0^{**}	.5202 (.5957)	60.0^{**}
		(4,2)	.4817	.5234 (.5289)	60.0^{**}	.5299 (-.4645)	50.7

Table 4. Out-of-Sample MSE and Sign Predictions from Selected Networks: Japanese Yen.

Network Type	Test Obs.	Selected Network	Recursive Result			NLS Result	
			PSC	MSE	Sign	MSE	Sign
Feed-forward	50	(1,6)	.4490	.1178 (.5929)	64.0**	.1125 (.5886)	60.0*
		(2,6)	.4532	.1181 (.6201)	64.0**	.1151 (.5495)	64.0**
		(4,2)	.4616	.1153 (.6045)	70.0**	.1156 (.5878)	66.0**
	100	(2,3)	.4732	.1721 (.4877)	50.0	.1791 (-.5285)	49.0
		(1,3)	.4752	.1701 (.5857)	59.0**	.1709 (.5739)	61.0**
		(4,2)	.4754	.1747 (-.4577)	54.0	.1765 (-.4741)	50.0
	150	(1,5)	.4785	.2293 (.4430)	58.0**	.2265 (.5662)	56.7**
		(1,3)	.4815	.2282 (.5111)	59.3**	.2320 (-.4858)	59.3**
		(1,4)	.4820	.2285 (.4945)	58.0**	.2281 (.5704)	57.3**
Recurrent	50	(1,4)	.4571	.1227 (-.4108)	40.0	.1150 (.6031)	66.0**
		(5,2)	.4624	.1124 (.6142)	54.0	.1076 (.5835)	56.0
		(5,4)	.4630	.1203 (.5569)	50.0	.1265 (-.5032)	52.0
	100	(1,3)	.4716	.1716 (.5313)	57.0*	.1729 (.5421)	55.0
		(4,2)	.4740	.1762 (-.5320)	51.0	.1872 (-.5783)	53.0
		(5,2)	.4749	.1804 (-.6110)	47.0	.1768 (-.4779)	56.0
	150	(1,3)	.4807	.2262 (.6184)	58.7**	.2312 (-.4673)	58.0**
		(5,4)	.4809	.2291 (.4608)	58.7**	.2495 (-.6037)	50.7
		(6,2)	.4810	.2345 (-.5697)	50.0	.2480 (-.5938)	50.7

Table 5. Out-of-Sample MSE and Sign Predictions from Selected Networks: Swiss Franc.

Network Type	Test Obs.	Selected Network	Recursive Result PSC	Recursive Result MSE	Recursive Result Sign	NLS Result MSE	NLS Result Sign
Feed-forward	50	(2,5)	.5743	.2039 (.6203)	66.0**	.2030 (.6117)	62.0**
		(3,3)	.5743	.2037 (.6104)	60.0*	.2009 (.5936)	66.0**
		(2,2)	.5748	.2069 (.6273)	62.0**	.1965 (.6182)	66.0**
	100	(2,4)	.5712	.4222 (-.4563)	55.0	.4482 (-.5653)	53.0
		(3,3)	.5718	.4151 (.5177)	58.0*	.4187 (.4243)	59.0**
		(2,2)	.5723	.4161 (.5626)	57.0*	.4185 (.4253)	54.0
	150	(2,5)	.5772	.4228 (-.4761)	58.0**	.4466 (-.5871)	56.7**
		(2,2)	.5785	.4163 (.5788)	58.7**	.4212 (-.4213)	58.7**
		(2,3)	.5789	.4132 (.5540)	57.3**	.4415 (-.5831)	50.0
Recurrent	50	(1,2)	.5720	.2062 (.6271)	62.0**	.2105 (.5569)	64.0**
		(1,4)	.5738	.2063 (.6009)	62.0**	.2061 (.5770)	64.0**
		(4,2)	.5750	.2098 (.5869)	64.0**	.2104 (.5235)	68.0**
	100	(1,2)	.5696	.4157 (.5560)	57.0*	.4205 (-.4251)	58.0**
		(1,4)	.5716	.4184 (.4586)	57.0*	.4468 (-.5754)	55.0
		(1,3)	.5725	.4155 (.5736)	57.0*	.4373 (-.5354)	57.0*
	150	(1,2)	.5770	.4148 (.5828)	58.7**	.4146 (.5302)	55.3*
		(1,4)	.5802	.4173 (.5166)	58.0**	.4170 (.5113)	58.0**
		(3,2)	.5806	.4200 (.3302)	56.0*	.4218 (-.4526)	58.0**

Table 6. Out-of-Sample MSE and Sign Predictions from ARMA Models.

Test Obs.	ARMA Model	BP MSE	BP Sign	CD MSE	CD Sign	DM MSE	DM Sign	JY MSE	JY Sign	SF MSE	SF Sign
50	(0,0)	.3884	N/A	.1907	N/A	.2099	N/A	.1225	N/A	.2157	N/A
	(1,0)	.3893	60.0*	.1890*	46.0	.2098	48.0	.1199	52.0	.2163	56.0
		(-.285)		(1.374)		(.023)		(.950)		(-.347)	
	(0,1)	.3897	62.0**	.1889*	46.0	.2099	48.0	.1202	50.0	.2163	54.0
		(-.380)		(1.428)		(-.004)		(.916)		(-.377)	
	(1,1)	.3915	56.0	.1885*	46.0	.2096	46.0	.1198	52.0	.2159	58.0
		(-.796)		(1.524)		(.144)		(.964)		(-.112)	
	(2,2)	.3910	58.0	.1896	44.0	.2034	54.0	.1202	50.0	.2124	52.0
		(-.686)		(1.071)		(1.100)		(.916)		(.859)	
100	(0,0)	.5730	N/A	.3119	N/A	.6055	N/A	.1740	N/A	.4198	N/A
	(1,0)	.5741	59.0**	.3144	43.0	.6027	48.0	.1708*	56.0	.4185	54.0
		(-.289)		(-1.015)		(.796)		(1.632)		(.573)	
	(0,1)	.5743	59.0**	.3144	44.0	.6030	48.0	.1708*	54.0	.4186	52.0
		(-.334)		(-.970)		(.788)		(1.624)		(.572)	
	(1,1)	.5745	56.0	.3145	44.0	.6028	47.0	.1709*	56.0	.4184	54.0
		(-.430)		(-.865)		(.840)		(1.625)		(.630)	
	(2,2)	.5814†	48.0	.3138	46.0	.6001	55.0	.1705**	53.0	.4190	53.0
		(-1.627)		(-1.027)		(.840)		(1.693)		(.163)	
150	(0,0)	.5016	N/A	.4158	N/A	.5273	N/A	.2299	N/A	.4200	N/A
	(1,0)	.5023	55.3*	.4200†	44.7	.5253	50.7	.2252**	54.7	.4204	49.3
		(-.234)		(-1.581)		(.934)		(1.826)		(-.174)	
	(0,1)	.5024	55.3*	.4199†	45.3	.5255	50.7	.2252**	52.7	.4203	48.0
		(-.276)		(-1.493)		(.934)		(1.855)		(-.137)	
	(1,1)	.5026	52.0	.4198†	45.3	.5253	50.7	.2254**	55.3*	.4209	50.7
		(-.363)		(-1.350)		(.935)		(1.764)		(-.444)	
	(2,2)	.5069†	47.3	.4198††	44.7	.5259	51.3	.2251**	54.7	.4269	49.3
		(-1.357)		(-1.756)		(.241)		(1.856)		(-1.083)	

Note: For MSE, † and †† (* and **) indicate the models that are significantly worse (better) than the random walk model at 10% and 5% level, respectively.

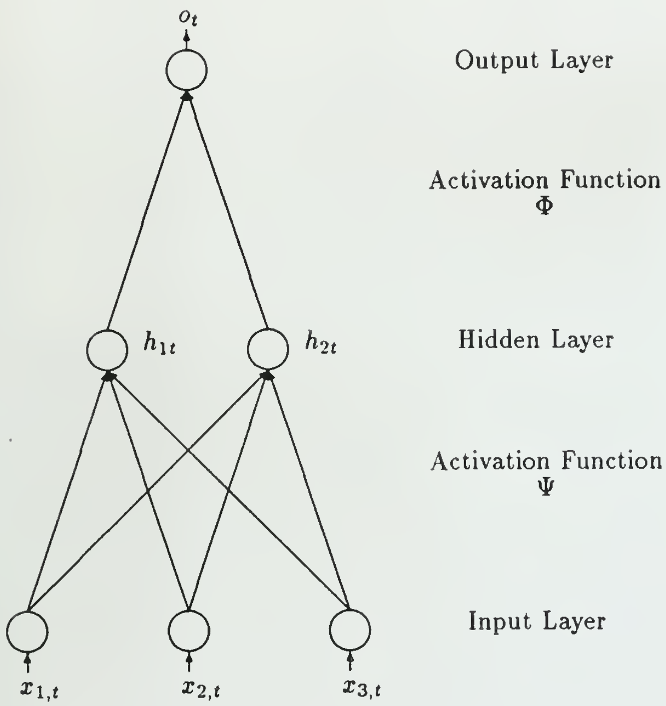

Figure 1: A Simple Feedforward Network with One Output Unit, Two Hidden Units, and Three Input Units.

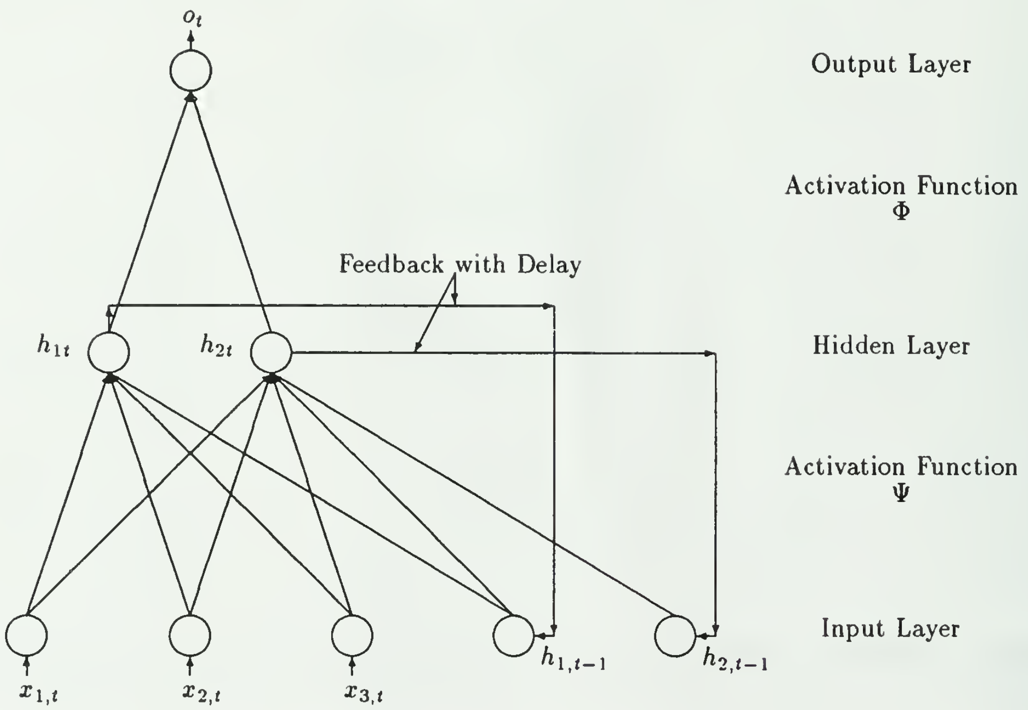

Figure 2: A Simple Elman (1990) Network with Hidden-Unit Activations Feedback.